

Amendments to the Claims

The following listing of claims will replace all prior versions and listings of claims in the application:

Listing of Claims:

1. **(Original)** A method of verifying a logic design for proper operation of tri-state buses specified in the design, the method comprising:
for each bus in the circuit design, performing an exhaustive analysis on a min-cut set of logic controlling the bus and designating each said bus as either conclusively
5 conflict-free and float-free or as inconclusive.

2. **(Original)** A method as defined in claim 1, further including performing a full exhaustive analysis of the bus when the exhaustive analysis on the min-cut set of logic is inconclusive.

3. (Original) A method as defined in claim 1, said performing an exhaustive analysis on a min-cut set further including:

determining a min-cut set of logic elements having the smallest size, said min-cut set

5 being a cut-set of logic elements which separates the control logic cone of the
bus into two parts in which all paths between a logic element in one part and a
logic element in the other part pass through an element in the min cut-set of
logic elements;

determining the inputs of the min-cut set of logic elements;

for each of a plurality of combinations of logic values:

10 assigning the combination of logic values to the inputs of the min-cut set of logic
elements;

forward simulating the logic values from said inputs to the select inputs of each
tri-state gate of the bus under test so as to determine the input of each
said tri-state to said bus;

15 determining whether a bus-conflict or a floating bus condition exists; and

designating said min-cut set exhaustive analysis as inconclusive
when a conflict or floating condition exists and

designating said bus as conflict-free and float-free when no conflict condition has been
determined after all of said plurality of combinations of logic values have been
20 evaluated.

4. (Original) A method as defined in claim 3 said assigning the combination of
logic values including generating a random set of logic values.

5. **(Original)** A method as defined in claim 2, said full exhaustive analysis comprising:
determining the inputs of the logic cone of the select inputs of the tri-state gates of the bus under test;
5 for each of a plurality of combinations of logic values:
 assigning the combination of logic values to said inputs of said logic cone;
 forward simulating said logic values from said logic cone inputs to the select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state gate to said bus;
10 determining whether a bus-conflict or a floating bus condition exists;
 designating said bus as a conflict bus when a bus-conflict or floating bus condition exists;
designating said bus as a conflict-free and float-free bus when no bus-conflict or floating bus condition has been determined after evaluating all of said plurality of
15 combinations of logic values.
6. **(Original)** A method as defined in claim 5, said assigning the combination of logic values including generating a random set of logic values.
7. **(Original)** A method as defined in claim 1, further including performing pre-analysis processing comprising:
identifying all buses in said circuit design; and
arranging identified buses in sorted list for processing in which buses in the fan-in of
5 other buses appear earlier in the list than the other buses.
8. **(Original)** A method as defined in claim 7, said pre-analysis processing including mapping any non-scanned pipeline flops as buffers before analyzing the buses.

9. **(Original)** A method as defined in claim 7, said pre-processing including:
applying constant logic values on one or more inputs of said circuit, including assigning
and propagating said constant logic values; and
maintaining constant throughout said analyses said constant logic values and logic
5 values resulting from said propagating.

10. **(Original)** A method as defined in claim 5, said forward simulating including
performing a parallel application of said combinations of logic values in which multiple
combinations are evaluated simultaneously by using each bit of an integer to store one
combination.

11. **(Original)** A method of verifying a logic design for proper operation of tri-state buses specified in the design, the method comprising:

identifying all buses in said circuit design;

5 arranging identified buses in a sorted list for processing in which buses in the fan-in of other buses appear earlier in the list than the other buses; and

for each bus in said list:

performing an exhaustive analysis on a min-cut set of logic controlling the bus, said performing an exhaustive analysis on a min-cut set further including:

10 determining a min-cut set of logic elements having the smallest size, said min-cut set being a cut-set of logic elements which separates the control logic cone of the bus into two parts in which all paths between a logic element in one part and a logic element in the other part pass through an element in the min-cut set of logic elements;

15 determining the inputs of the min-cut set of logic elements; and for each of a plurality of combinations of logic values:

assigning the combination of logic values to the inputs of said min-cut set of logic elements;

20 forward simulating the logic values from said inputs to the select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state to said bus;

determining whether a bus-conflict or a floating bus condition exists; and, if so, designating said min-cut set exhaustive analysis as inconclusive and performing a full exhaustive analysis of the bus; and

25 designating said bus as conflict free and float-free when no conflict conditions have been determined after evaluating all of said combinations of logic values.

12. (Original) A method as defined in claim **11**, said forward simulating including performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

13. (Original) A method as defined in claim **11**, said full exhaustive analysis comprising:

determining the inputs of the logic cone of the select inputs of the tri-state gates of the bus under test;

5 for each of a plurality of combinations of logic values:

assigning the combination of logic values to said inputs of the logic cone;

forward simulating the logic values from said logic cone inputs to the select

inputs of each tri-state gate of the bus under test so as to determine the

input logic value of each said tri-state gate to said bus;

10 determining whether a bus-conflict or a floating bus condition exists;

designating said bus as a conflict bus when a conflict or floating condition exists;

designating said bus as a conflict-free and float-free bus when no bus-conflict or

floating condition has been determined after evaluating all of said plurality of

15 combinations of logic values.

14. (Original) A method as defined in claim **13**, said forward simulating including performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

- 15. (Original)** A method as defined in claim 1, further including, prior to performing an exhaustive analysis on a min-cut set of logic controlling the bus:
performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free;
5 performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free; and
designating said bus as a no-conflict bus when said bus is determined to be conflict-free and floating free and selecting a next bus in sequence for analysis.

16. (Original) A method as defined in claim **15**, said performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free, comprising:

for each input of the bus:

- 5 implying logic values to signals which control the control input of the tri-state gate associated with said each input so as to produce an enabling control value;
- determining whether said implying logic values results in an implication conflict; when an implication conflict exists, selecting another of said inputs and
- 10 repeating said implying and determining steps;
- when an implication conflict does not exist, determining whether said implying logic values results in a bus conflict;
- terminating said implication based conflict-free analysis and performing a full exhaustive analysis when a
- 15 bus conflict is determined;
- determining whether all other inputs to said bus are in a high impedance state and, if not, terminating said implication based conflict-free analysis and performing an exhaustive analysis on a min-cut set of logic controlling the bus; and
- 20 designating said bus as conclusively conflict-free when each said implying logic values resulted in either an implication conflict all other inputs to said bus being in a high impedance state.

17. (Original) A method as defined in claim **15**, said performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free, comprising:

implying logic values to signals which control the control input of each tri-state gate

5 associated with said bus so as to produce disabling control value on the control inputs of said gates;

determining whether said implying logic values results in an implication conflict;

terminating said implication based floating-free analysis and performing an exhaustive analysis on a min-cut set of logic controlling the bus when no implication conflict

10 is determined; and

designating said bus as floating-free when an implication conflict is determined.

- 18. (Original)** A method as defined in claim 17, said performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free, comprising:
- for each input of said bus which is controlled by a tri-state gate having a select input;
- 5 initializing the circuit to unknown values;
- assigning an enabling value to the select input of the bus;
- implying logic values to inputs to said gate needed to produce an enabling value;
- determining whether said implying logic values results in an implication conflict and
- selecting a next input for analysis when an implication conflict is detected;
- 10 determining whether a bus conflict exists at the inputs of said bus and
- performing a full exhaustive analysis of said bus when a bus conflict exists;
- when no bus-conflict exists,
- determining whether all other inputs of said bus are in a high-
- 15 impedance state and, if not, performing said exhaustive analysis on a min-cut set of logic controlling the bus, and, if so, selecting a next input for analysis; and
- designating said bus as conclusively conflict-free when each said implying logic values resulted in either an implication conflict all other inputs to said bus being in a high
- 20 impedance state.

- 19. (Original)** A method of verifying a logic design for proper operation of tri-state buses specified in the design, the method comprising, for each bus in the circuit design: performing an implication based conflict-free analysis on said each bus to determine whether said bus is conflict-free;
- 5 performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free; and
- designating said bus as a no-conflict bus when said each bus is determined to be conflict-free and floating free; and
- performing an exhaustive analysis of the bus when either of said implication based
- 10 analyses is inconclusive.

20. (Original) A method as defined in claim **19**, said performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free, comprising:

for each input of the bus:

5 implying logic values to signals which control the control input of the tri-state gate associated with said each input so as to produce an enabling control value;

 determining whether said implying logic values results in an implication conflict;

 when an implication conflict exists, selecting another of said inputs and

10 repeating said implying and determining steps;

 when an implication conflict does not exist, determining whether said

 implying logic values results in a bus conflict and terminating said

 implication based conflict-free analysis and performing a full

 exhaustive analysis when a bus conflict is determined;

15 determining whether all other inputs to said bus are in a high impedance state

 and, if not, terminating said implication based conflict-free analysis and

 performing an exhaustive analysis on a min-cut set of logic controlling the

 bus; and

 designating said bus as conclusively conflict-free when each said implying logic values

20 resulted in either an implication conflict all other inputs to said bus being in a high impedance state.

21. (Original) A method as defined in claim **19**, said performing an implication based, floating-free analysis on said bus to determine whether said bus is floating-free, comprising:

implying logic values to signals which control the control input of each tri-state gate

5 associated with said bus so as to produce disabling control value on the control inputs of said gates;

determining whether said implying logic values produces an implication conflict;

terminating said implication based floating-free analysis and performing an exhaustive analysis on said bus when no implication conflict is

10 determined; and

designating said bus as floating-free when an implication conflict is determined.

22. (Original) A method as defined in claim **20**, said performing an implication based, floating-free analysis on said bus to determine whether said bus is floating-free, comprising:

implying logic values to signals which control the control input of each tri-state gate

5 associated with said bus so as to produce disabling control value on the control inputs of said gates;

determining whether said implying logic values produces an implication conflict;

terminating said implication based floating-free analysis and performing an exhaustive analysis on a min-cut set of logic controlling the bus when no

10 implication conflict is determined; and

designating said bus as floating-free when an implication conflict is determined.

23. (Original) A method as defined in claim **19**, said exhaustive analysis including an exhaustive analysis on a min-cut set of logic controlling the bus.

24. (Original) A method as defined in claim **23**, said exhaustive analysis on a min-cut set including:

determining a min-cut set of logic elements having the smallest size, said min-cut set being a cut-set of logic elements which separates the control logic cone of the bus into two parts in which all paths between a logic element in one part and a logic element in the other part pass through an element in the min cut-set of logic elements;

determining the inputs of the min-cut set of logic elements;

for each of a plurality of combinations of logic values:

assigning the combination of logic values to the inputs of the min-cut set of logic elements;

forward simulating the logic values from said inputs to the select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state to said bus;

determining whether a bus-conflict or a floating bus condition exists; and designating said min-cut set exhaustive analysis as inconclusive when a when a conflict or floating condition exists and

designating said bus as conflict-free and float-free when no conflict condition has been determined after all of said plurality of combinations of logic values have been evaluated.

25. (Original) A method as defined in claim **24**, said forward simulating including performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

26. (Original) A method as defined in claim **23**, said exhaustive analysis being a full exhaustive analysis including:

determining the inputs of the logic cone of the select inputs of the tri-state gates of the bus under test;

5 for each of a plurality of combinations of logic values:

assigning the combination of logic values to said inputs of the logic cone;

forward simulating the logic values from said logic cone inputs to the select

inputs of each tri-state gate of the bus under test so as to determine the

input of each said tri-state gate to said bus;

10 determining whether a bus-conflict or a floating bus condition exists; and

designating said bus as having a no conflict bus when no conflict or floating condition

exists and designating said bus as a conflict bus when a conflict or floating

condition exists.

- 27. (Original)** A method of verifying a logic design for proper operation of tri-state buses specified in the design, the method comprising, for each bus in the circuit design: performing an implication based conflict-free analysis on said each bus to determine whether said bus is conflict-free; performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free; and designating said bus as a no-conflict bus when said bus is determined to be conflict-free and floating-free and performing an exhaustive analysis of the bus when either of said implication based analyses is inconclusive.
- 28. (Original)** A method as defined in claim **27**, said performing an exhaustive analysis including an exhaustive analysis on a min-cut set of logic controlling the bus.
- 29. (Original)** A method as defined in claim **28**, said performing an exhaustive analysis including performing a full exhaustive analysis of the bus when the exhaustive analysis on the min-cut set of logic is inconclusive.

30. (Original) A method as defined in claim **29**, said exhaustive analysis on a min-cut set including:

determining a min-cut set of logic elements having the smallest size, said min-cut set being a cut-set of logic elements which separates the control logic cone of the bus into two parts in which all paths between a logic element in one part and a logic element in the other part pass through an element in the min cut-set of logic elements;

determining the inputs of the min-cut set of logic elements;

for each of a plurality of combinations of logic values:

10 assigning the combination of logic values to the inputs of the min-cut set of logic elements;

 forward simulating the logic values from said inputs to the select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state to said bus;

15 determining whether a bus-conflict or a floating bus condition exists; and
 designating said min-cut set exhaustive analysis as inconclusive when a
 when a conflict or floating condition exists and

 designating said bus as conflict-free and float-free when no conflict condition has
 been determined after all of said plurality of combinations of logic values
20 have been evaluated.

31. (Original) A method as defined in claim **30**, said performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free comprising:

for each input of the bus:

- 5 implying logic values to signals which control the control input of the tri-state gate associated with said each input so as to produce an enabling control value;
- determining whether said implying logic values results in an implication conflict; when an implication conflict exists, selecting another of said inputs and
- 10 repeating said implying and determining steps;
- when an implication conflict does not exist, determining whether said implying logic values results in a bus conflict;
- terminating said implication based conflict-free analysis and performing a full exhaustive analysis when a
- 15 bus conflict is determined;
- determining whether all other inputs to said bus are in a high impedance state and, if not, terminating said implication based conflict-free analysis and performing said exhaustive analysis on a min-cut set of logic controlling the bus; and
- 20 designating said bus as conclusively conflict-free when each said implying logic values resulted in either an implication conflict all other inputs to said bus being in a high impedance state.

32. (Original) A method as defined in claim **31**, said performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free comprising:

initializing the circuit to unknown values;

- 5 implying logic values to signals which control the control input of each tri-state gate associated with said bus so as to produce disabling control value on the control inputs of said gates;

determining whether said implying logic values produces an implication conflict;

- 10 terminating said implication based floating-free analysis and performing an exhaustive analysis on a min-cut set of logic controlling the bus when no implication conflict is determined; and

designating said bus as floating-free when an implication conflict is determined.

33. (Original) A method as defined in claim **32**, said full exhaustive analysis including:

determining the inputs of the logic cone of the select inputs of the tri-state gates of the bus under test;

- 5 for each of a plurality of combinations of logic values:

assigning the combination of logic values to said inputs of the logic cone;

forward simulating the logic values from said logic cone inputs to the select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state gate to said bus;

- 10 determining whether a bus-conflict or a floating bus condition exists;

designating said bus as a conflict bus when a conflict or floating condition exists;

designating said bus as a conflict-free and float-free bus when no bus-conflict or floating condition has been determined after evaluating all of said plurality of combinations of logic values.

34. (Original) A method as defined in claim **33**, said assigning the combination of logic values including generating a random set of logic values.

35. (Original) A method as defined in claim **34**, further including performing pre-analysis processing comprising:

identifying all buses in said circuit design; and

arranging identified buses in sorted list for processing in which buses in the fan-in of

5

other buses appear earlier in the list than the other buses.

36. (Original) A method as defined in claim **35**, said pre-analysis processing including mapping any non-scanned pipeline flops as buffers before analyzing the buses.

37. (Original) A method as defined in claim **36**, said forward simulating including performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

38. (Original) A method as defined in claim **37**, said pre-processing including: applying of constant logic values on one or more inputs of said circuit, including

assigning and propagating said constant logic values; and

maintaining constant throughout said analyses said constant logic values and logic

5

values resulting from said propagating.

39. (Original) A program product for use in verifying a circuit logic design for proper operation of tri-state buses specified in the design, the program product comprising:
a computer readable storage medium;

5 means recorded on the medium for, for each bus in the circuit design, performing an exhaustive analysis on a min-cut set of logic controlling the bus and designating each said bus as either conclusively conflict-free and float-free or as inconclusive.

40. (Original) A program product as defined in claim **39**, further including means
10 recorded on said medium for performing a full exhaustive analysis of the bus when the exhaustive analysis on the min-cut set of logic is inconclusive.

41. (Original) A program product as defined in claim **39**, said means for performing an exhaustive analysis on a min-cut set further including:

means recorded on the medium for determining a min-cut set of logic elements having

the smallest size, said min-cut set being a cut-set of logic elements which

5 separates the control logic cone of the bus into two parts in which all paths between a logic element in one part and a logic element in the other part pass through an element in the min cut-set of logic elements;

means recorded on the medium for determining the inputs of the min-cut set of logic elements;

10 means recorded on the medium for, for each of a plurality of combinations of logic values:

assigning the combination of logic values to the inputs of the min-cut set of logic elements;

forward simulating the logic values from said inputs to the select inputs of each

15 tri-state gate of the bus under test so as to determine the input of each said tri-state to said bus;

determining whether a bus-conflict or a floating bus condition exists; and

designating said min-cut set exhaustive analysis as inconclusive when a conflict or floating condition exists and

20 means recorded on said medium for designating said bus as conflict-free and float-free when no conflict condition has been determined after all of said plurality of combinations of logic values have been evaluated.

42. (Original) A program product as defined in claim **41** said means for assigning the combination of logic values including means recorded on said medium for generating a random set of logic values.

43. (Original) A program product as defined in claim **40**, said full exhaustive analysis comprising:

means recorded on said medium for determining the inputs of the logic cone of the select inputs of the tri-state gates of the bus under test;

5 means recorded on said medium for, for each of a plurality of combinations of logic values:

means recorded on said medium for assigning the combination of logic values to said inputs of said logic cone;

10 means recorded on said medium for forward simulating said logic values from said logic cone inputs to the select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state gate to said bus;

means recorded on said medium for determining whether a bus-conflict or a floating bus condition exists;

15 means recorded on said medium for designating said bus as a conflict bus when a bus-conflict or floating bus condition exists;

means recorded on said medium for designating said bus as a conflict-free and float-free bus when no bus-conflict or floating bus condition has been determined after evaluating all of said plurality of combinations of logic values.

44. (Original) A program product as defined in claim **43**, said means for assigning the combination of logic values including means recorded on said medium for generating a random set of logic values.

45. (Original) A program product as defined in claim **39**, further including means recorded on said medium for performing pre-analysis processing comprising:

means recorded on said medium for identifying all buses in said circuit design;

and

5 arranging identified buses in sorted list for processing in which buses in the fan-in of other buses appear earlier in the list than the other buses.

46. (Original) A program product as defined in claim **45**, said means for performing pre-analysis processing including means recorded on said medium for mapping any non-scanned pipeline flops as buffers before analyzing the buses.

47. (Original) A program product as defined in claim **45**, said means for performing pre-processing including:

means recorded on said medium for applying constant logic values on one or more inputs of said circuit, including assigning and propagating said constant logic values; and

5

means recorded on said medium for maintaining constant throughout said analyses said constant logic values and logic values resulting from said propagating.

48. (Original) A program product as defined in claim **41**, said means for forward simulating including performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

49. (Original) A program product for use in verifying a logic design for proper operation of tri-state buses specified in the design, the program product comprising: a computer readable storage medium;

means recorded on said medium for identifying all buses in said circuit design;

5 means recorded on said medium for arranging identified buses in a sorted list for processing in which buses in the fan-in of other buses appear earlier in the list than the other buses; and

means recorded on said medium for performing an exhaustive analysis on a min-cut set of logic controlling the bus, said means for performing an exhaustive analysis on
10 a min-cut set further including:

means recorded on said medium for determining a min-cut set of logic elements having the smallest size, said min-cut set being a cut-set of logic elements which separates the control logic cone of the bus into two parts in which all paths between a logic element in
15 one part and a logic element in the other part pass through an element in the min cut-set of logic elements;

means recorded on said medium for determining the inputs of the min-cut set of logic elements; and

means recorded on said medium for, for each of a plurality of combinations of logic values:

20 assigning the combination of logic values to the inputs of said min-cut set of logic elements;

forward simulating the logic values from said inputs to the select inputs of each tri-state gate of the bus under test so as to
25 determine the input of each said tri-state gate to said bus;

determining whether a bus-conflict or a floating bus condition exists; and, if so, designating said min-cut set exhaustive analysis as inconclusive and performing a full exhaustive analysis of the bus;

30 means recorded on said medium for designating said bus as conflict free

and float-free when no conflict conditions have been determined after evaluating all of said combinations of logic values.

50. (Original) A program product as defined in claim **49**, said means for forward simulating including means recorded on said medium for performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

51. (Original) A program product as defined in claim **49**, said means for performing a full exhaustive analysis comprising:

means recorded on said medium for determining the inputs of the logic cone of the select inputs of the tri-state gates of the bus under test;

5 means recorded on said medium for, for each of a plurality of combinations of logic values:

assigning the combination of logic values to said inputs of the logic cone;

forward simulating the logic values from said logic cone inputs to the select

10 inputs of each tri-state gate of the bus under test so as to determine the

input of each said tri-state gate to said bus;

determining whether a bus-conflict or a floating bus condition exists;

designating said bus as a conflict bus when a conflict or floating condition exists;

means recorded on said medium for designating said bus as a conflict-free and

15 float-free bus when no bus-conflict or floating condition has been

determined after evaluating all of said plurality of combinations of logic values.

52. (Original) A program product as defined in claim **51**, said means for forward simulating including means recorded on said medium for performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

- 53. (Original)** A program product as defined in claim **39**, further including, prior to performing an exhaustive analysis on a min-cut set of logic controlling the bus:
means recorded on said medium for performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free;
- 5 means recorded on said medium for performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free; and
means recorded on said medium for designating said bus as a no-conflict bus when said bus is determined to be conflict-free and floating free and selecting a next bus in sequence for analysis.

54. (Original) A program product as defined in claim **53**, said means for performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free, comprising:

means recorded on said medium for implying logic values to signals which control the

5 control input of the tri-state gate associated with said each input so as to
 produce an enabling control value;

means recorded on said medium for determining whether said implying logic values
 results in an implication conflict;

 when an implication conflict exists, selecting another of said inputs and
10 repeating said means for implying and said means for
 determining;

 when an implication conflict does not exist, determining whether said
 implying logic values results in a bus conflict;

 terminating said implication based conflict-free analysis
15 and performing a full exhaustive analysis when a
 bus conflict is determined;

means recorded on said medium for determining whether all other inputs to said bus are
 in a high impedance state and, if not, terminating said implication based conflict-
 free analysis and performing an exhaustive analysis on a min-cut set of logic
20 controlling the bus; and

means recorded on said medium for designating said bus as being conflict-free when an
 implication conflict has been detected and as a bus conflict when an implication
 conflict has not been detected on said inputs of said bus.

55. (Original) A program product as defined in claim 53, said means for performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free, comprising:

means recorded on said medium for implying logic values to signals which control the

5 control input of each tri-state gate associated with said bus so as to produce
 disabling control value on the control inputs of said gates;

means recorded on said medium for determining whether said implying logic values
 results in an implication conflict;

 terminating said implication based floating-free analysis and performing an
10 exhaustive analysis on a min-cut set of logic controlling the bus when no
 implication conflict is determined; and

means recorded on said medium for designating said bus as floating-free when an
 implication conflict is determined.

56. (Original) A program product as defined in claim **53**, said means for performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free, comprising:

means recorded on said medium for, for each input of said bus which is controlled by a

5 tri-state gate having a select input;

initializing the circuit to unknown values;

assigning an enabling value to the select input of the bus;

implying logic values to inputs to said gate needed to produce an enabling value;

determining whether said implying logic values results in an implication conflict and

10 selecting a next input for analysis when an implication conflict is detected;

determining whether a bus conflict exists at the inputs of said bus and

performing a full exhaustive analysis of said bus when a bus conflict

exists;

when no bus-conflict exists:

15 determining whether all other inputs of said bus are in a high-

impedance state and, if not, performing said exhaustive

analysis on a min-cut set of logic controlling the bus, and,

if so, selecting a next input for analysis; and

means recorded on said medium for designating said bus as conclusively conflict-free

20 when each said implying logic values resulted in either an implication conflict all

other inputs to said bus being in a high impedance state.

- 57. (Original)** A program product for use in verifying a logic design for proper operation of tri-state buses specified in the design, the program product comprising: means recorded on said medium for performing an implication based conflict-free analysis on said each bus to determine whether said bus is conflict-free;
- 5 means recorded on said medium for performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free;
- means recorded on said medium for designating said bus as a no-conflict bus when said each bus is determined to be conflict-free and floating free; and
- 10 means recorded on said medium for performing an exhaustive analysis of the bus when either of said implication based analyses is inconclusive.

58. (Original) A program product as defined in claim **57**, said means for performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free, comprising:

means recorded on said medium for, for each input of the bus:

5 implying logic values to signals which control the control input of the tri-state gate associated with said each input so as to produce an enabling control value;

 determining whether said implying logic values results in an implication conflict;

 when an implication conflict exists, selecting another of said inputs and

10 repeating said implying and determining steps;

 when an implication conflict does not exist, determining whether said

 implying logic values results in a bus conflict and terminating said

 implication based conflict-free analysis and performing a full

 exhaustive analysis when a bus conflict is determined;

15 determining whether all other inputs to said bus are in a high impedance state

 and, if not, terminating said implication based conflict-free analysis and

 performing an exhaustive analysis on a min-cut set of logic controlling the

 bus; and

means recorded on said medium for designating said bus as conclusively conflict-free

20 when each said implying logic values resulted in either an implication conflict all other inputs to said bus being in a high impedance state.

59. (Original) A program product as defined in claim 57, said means for performing an implication based, floating-free analysis on said bus to determine whether said bus is floating-free, comprising:

means recorded on said medium for implying logic values to signals which control the

5 control input of each tri-state gate associated with said bus so as to produce
 disabling control value on the control inputs of said gates;

means recorded on said medium for determining whether said implying logic values
 produces an implication conflict;

means recorded on said medium for terminating said implication based floating-free
10 analysis and performing an exhaustive analysis on said bus when no implication
 conflict is determined; and

means recorded on said medium for designating said bus as floating-free when an
 implication conflict is determined.

60. (Original) A program product as defined in claim 58, said means for performing an implication based, floating-free analysis on said bus to determine whether said bus is floating-free, comprising:

means recorded on said medium for implying logic values to signals which control the

5 control input of each tri-state gate associated with said bus so as to produce
 disabling control value on the control inputs of said gates;

means recorded on said medium for determining whether said implying logic values
 produces an implication conflict;

means recorded on said medium for terminating said implication based floating-free
10 analysis and performing an exhaustive analysis on a min-cut set of logic
 controlling the bus when no implication conflict is determined; and

means recorded on said medium for designating said bus as floating-free when an
 implication conflict is determined.

61. (Original) A program product as defined in claim **57**, said means for performing an exhaustive analysis including means recorded on said medium for performing an exhaustive analysis on a min-cut set of logic controlling the bus.

62. (Original) A program product as defined in claim **61**, said means for performing an exhaustive analysis on a min-cut set including:

means recorded on said medium for determining a min-cut set of logic elements having

the smallest size, said min-cut set being a cut-set of logic elements which

5 separates the control logic cone of the bus into two parts in which all paths between a logic element in one part and a logic element in the other part pass through an element in the min cut-set of logic elements;

means recorded on said medium for determining the inputs of the min-cut set of logic elements;

10 means recorded on said medium for, for each of a plurality of combinations of logic values:

assigning the combination of logic values to the inputs of the min-cut set of logic elements;

forward simulating the logic values from said inputs to the select inputs of each

15 tri-state gate of the bus under test so as to determine the input of each said tri-state to said bus;

determining whether a bus-conflict or a floating bus condition exists; and

designating said min-cut set exhaustive analysis as inconclusive when a conflict condition exists and

20 means recorded on said medium for designating said bus as conflict-free and float-free when no conflict condition has been determined after all of said plurality of combinations of logic values have been evaluated.

63. (Original) A program product as defined in claim **61**, said means for forward simulating including means recorded on said medium for performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

64. (Original) A program product as defined in claim **61**, said means for performing an exhaustive analysis being a means for performing a full exhaustive analysis including:

means recorded on said medium for determining the inputs of the logic cone of the

5 select inputs of the tri-state gates of the bus under test;

means recorded on said medium for, for each of a plurality of combinations of logic values:

assigning the combination of logic values to said inputs of the logic cone;

forward simulating the logic values from said logic cone inputs to the

10 select inputs of each tri-state gate of the bus under test so as to determine the input of each said tri-state gate to said bus;

determining whether a bus-conflict or a floating bus condition exists; and

designating said bus as having a no conflict bus when no conflict

or floating condition exists and designating said bus as a

15 conflict bus when a conflict or floating condition exists.

65. (Original) A program product for use in verifying a logic design for proper operation of tri-state buses specified in the design, the program product comprising, for each bus in the circuit design:

means recorded on said medium for performing an implication based conflict-free

5 analysis on said each bus to determine whether said bus is conflict-free;

means recorded on said medium for performing an implication based floating-free

analysis on said bus to determine whether said bus is floating-free; and

means recorded on said medium for designating said bus as a no-conflict bus when said bus is determined to be conflict-free and floating-free and performing an

10 exhaustive analysis of the bus when either of said implication based analyses is inconclusive.

66. (Original) A program product as defined in claim **65**, said means for performing an exhaustive analysis including means recorded on said medium for performing an exhaustive analysis on a min-cut set of logic controlling the bus.

67. (Original) A program product as defined in claim **66**, said means for performing an exhaustive analysis including means recorded on said medium for performing a full exhaustive analysis of the bus when the exhaustive analysis on the min-cut set of logic is inconclusive.

68. (Original) A program product as defined in claim **67**, said means for performing an exhaustive analysis on a min-cut set including:

means recorded on said medium for determining a min-cut set of logic elements having

the smallest size, said min-cut set being a cut-set of logic elements which

5 separates the control logic cone of the bus into two parts in which all paths between a logic element in one part and a logic element in the other part pass through an element in the min cut-set of logic elements;

means recorded on said medium for determining the inputs of the min-cut set of logic elements;

10 means recorded on said medium for, for each of a plurality of combinations of logic values:

assigning the combination of logic values to the inputs of the min-cut set of logic elements;

forward simulating the logic values from said inputs to the select inputs of each

15 tri-state gate of the bus under test so as to determine the input of each said tri-state to said bus;

determining whether a bus-conflict or a floating bus condition exists; and

designating said min-cut set exhaustive analysis as inconclusive when a conflict or floating condition exists and

20 designating said bus as conflict-free and float-free when no conflict condition has been determined after all of said plurality of combinations of logic values have been evaluated.

69. (Original) A program product as defined in claim **68**, said means for performing an implication based conflict-free analysis on said bus to determine whether said bus is conflict-free comprising:

means recorded on said medium for, for each input of the bus:

5 implying logic values to signals which control the control input of the tri-state gate associated with said each input so as to produce an enabling control value;

 determining whether said implying logic values results in an implication conflict;

 when an implication conflict exists, selecting another of said inputs and

10 repeating said implying and determining steps;

 when an implication conflict does not exist, determining whether said implying logic values results in a bus conflict;

 terminating said implication based conflict-free analysis

 and performing a full exhaustive analysis when a

15 bus conflict is determined;

 determining whether all other inputs to said bus are in a high impedance state

 and, if not, terminating said implication based conflict-free analysis and

 performing said exhaustive analysis on a min-cut set of logic controlling the bus; and

20 designating said bus as conclusively conflict-free when each said implying logic values resulted in either an implication conflict all other inputs to said bus being in a high impedance state.

70. (Original) A program product as defined in claim **69**, said means for performing an implication based floating-free analysis on said bus to determine whether said bus is floating-free comprising:

means recorded on said medium for initializing the circuit to unknown values;

- 5 means recorded on said medium for implying logic values to signals which control the control input of each tri-state gate associated with said bus so as to produce disabling control value on the control inputs of said gates;

means recorded on said medium for determining whether said implying logic values produces an implication conflict;

- 10 means recorded on said medium for terminating said implication based floating-free analysis and performing an exhaustive analysis on a min-cut set of logic controlling the bus when no implication conflict is determined; and

means recorded on said medium for designating said bus as floating-free when an implication conflict is determined.

71. (Original) A program product as defined in claim **70**, said means for performing an full exhaustive analysis including:

means recorded on said medium for determining the inputs of the logic cone of the
select inputs of the tri-state gates of the bus under test;

5 means recorded on said medium for, for each of a plurality of combinations of logic
values:

assigning the combination of logic values to said inputs of the logic cone;

forward simulating the logic values from said logic cone inputs to the select
inputs of each tri-state gate of the bus under test so as to determine the
input of each said tri-state gate to said bus;

10

determining whether a bus-conflict or a floating bus condition exists;

designating said bus as a conflict bus when a conflict or floating condition
exists;

designating said bus as a conflict-free and float-free bus when no bus-conflict or
floating condition has been determined after evaluating all of said plurality
of combinations of logic values.

15

72. (Original) A program product as defined in claim **71**, said assigning the
combination of logic values including generating a random set of logic values.

73. (Original) A program product as defined in claim **72**, further including means
recorded on said medium for performing pre-analysis processing comprising:

identifying all buses in said circuit design; and

arranging identified buses in sorted list for processing in which buses in the fan-
in of other buses appear earlier in the list than the other buses.

5

74. (Original) A program product as defined in claim **73**, said pre-analysis
processing including means recorded on said medium for mapping any non-scanned
pipeline flops as buffers before analyzing the buses.

75. (Original) A program product as defined in claim **74**, said forward simulating including means recorded on said medium for performing a parallel application of said combinations of logic values in which multiple combinations are evaluated simultaneously by using each bit of an integer to store one combination.

76. (Original) A program product as defined in claim **75**, said means for pre-processing including:

means recorded on said medium for applying of constant logic values on one or more inputs of said circuit, including assigning and propagating said constant logic values; and

5

means recorded on said medium for maintaining constant throughout said analyses said constant logic values and logic values resulting from said propagating.